Evolutionary Optimization Methods for Accelerator Design

Alexey A. Poklonskiy

22 May 2008

Outline

- Introduction to Evolutionary Algorithms
- Applications in Accelerator Design
  - Quadrupole Triplet Telescope Design
  - Normal Form Defect Function Optimization
  - Neutrino Factory Front End Design

General Problem Formulation and Classification

Optimization is important!

- Define $f : S \longmapsto R$ — objective function, $\mathbf{x}$ — vector of control parameters
- Find $f^* \in R$, $\mathbf{x}^* \in S$:

$$f^* = f(\mathbf{x}^*) = \min_{\mathbf{x} \in S} f(\mathbf{x})$$

- Classification:
  - *Parameter types:* on/off, discrete, continuous, functions of a certain type, etc.
  - *Dimensionality:* number of control parameters
  - *Objective function number:* single and multi-objective
  - *Presence of constraints:* constrained and unconstrained
  - *Presence of noise:* noise could be present in parameters and in the objective function values
  - *Properties of the objective function:* modality, convexity, time-dependence, continuity, differentiability, smoothness, separability, etc

What is Evolutionary Algorithm?

- *Family:* heuristic, stochastic methods
- *Inspiration:* computational analogy of the **adaptive systems** from nature based on the principle of the **evolution** via a natural **selection** (C.Darwin, 1859)
- *Idea:* **population** of individuals undergoes selection in the presence of variation-inducing operators such as **mutation** and **recombination** (**crossover**). The **fitness function** is used to evaluate individuals. Reproductive success varies with fitness
- *Applicability:* does not guarantee the best solution, but often finds it or at least with a **partially optimal** solution (**good fit**). **Not a rigorous method**, but good in practice!

Evolutionary Algorithm (General Form)

```
Generate initial population, evaluate fitness
While stop condition not satisfied do
    Produce next population by
        Selection
        Recombination
    Evaluate fitness
End while
```

Why

1. Can be extended to constrained optimization
2. Capable of both exploration (broad search) and exploitation (local search)
3. In practice often find global extrema
4. Can generate/find unforeseen solutions (artificial design)
5. For multi-objective problems, return a set of satisfactory solutions. Useful to approximate Paretto front
6. Well suited for supporting design and optimization phases of decision making
7. Moderate computational cost
8. Relative simplicity of technical implementation and modification
9. Demonstrated record of successful applications

How and When

To design or select an EA for the problem:

1. **Effectively** encode solutions of a given problem to chromosomes in EA.
2. **Meaningfully** compare the relative performance (fitness) of solutions.

EAs are useful and efficient when

1. The search space is large, complex or poorly understood
2. Domain knowledge is scarce or expert knowledge is difficult to encode to narrow the search space
3. Objective function does not possess any “nice” properties, analytic tools are not applicable
4. Traditional search methods fail or are prohibitively computationally expensive

How in More Details

1. Select EA flavour.
2. Define a representation:
   - real number 1D, 2D, and 3D arrays
   - 1D, 2D, and 3D binary strings
   - lists
   - trees
3. Define genetic operators:
   - Crossover
   - Mutation
4. Define the objective function.
5. Set the algorithm parameters (probabilities, rates, thresholds, flags).

In reality steps are interconnected!

Critical Factors

- Might need extensive fine-tuning for the problem
- Need to keep evolutionary pressure in balance (similar to annealing schedule for Simulated Annealing)
- Possibility to choose a “right” representation but “wrong” genetic operator or vice versa

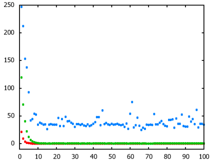
Record of Successful Applications

- *Optimization:* numerical optimization, combinatorial optimization problems (TSP), circuit design, timetabling and scheduling, video and sound quality optimization, optimal molecule configurations
- *Automatic Programming or Evolutionary Computing:* evolving computer programs for specific tasks (also filters for particle collision experiments), cellular automates, sorting networks
- *Machine and Robot Learning:* classification and prediction, neural networks, evolving rules for learning classifier systems and symbolic production systems, design and control in robotics
- *Economics:* modelling processes of innovation, the development of bidding strategies
- *Ecology and Biology:* biological arms races, host-parasite co-evolution, symbiosis and resource flow in nature, configuration applications, protein folding and protein/ligand docking (GARAGe)
- *Artificial Life:* evolution of intelligence and cooperation

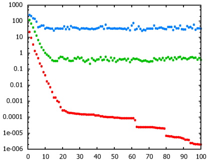
Why Do They Work?

- GA: John Holland, 1995, “Adaptation in Natural and Artificial Systems”: sampling hyperplane partitions in search space (being implemented properly)
- ES: Günter Rudolf, 1997, “Convergence Properties of Evolutionary Algorithms”: modelling EAs with Markov chains, convergence to global optimum can be proven assuming infinite time if elitism is in place; convergence speed needs additional assumptions about objective function

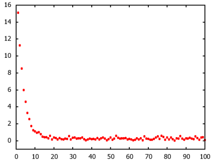
Example Statistics



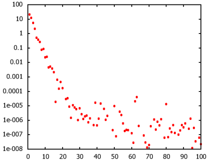
(a) Max/avg/min function values, normal axis



(b) Max/avg/min function values, logarithmic axis



(c) Estimated average Euclidean distance between population members



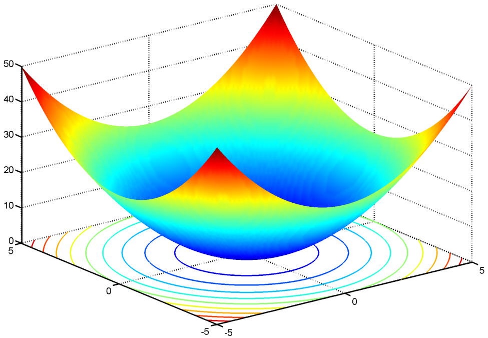
(d) Min function value improvement (absolute value)

GATool Algorithm

```
Randomly generate initial population, set predefined members, if any
Calculate objective function values, scale to fitnesses
Update statistics
While any of the stop conditions is not satisfied do
    Perform Roulette Wheel/Stochastic Uniform/Tournament Selection
    Generate next population
        Produce mutants by Uniform/Gaussian Mutation
        Produce children by Continuous Crossover
        Copy elite members
    Replace old population with newly generated
    Calculate objective function values, scale to fitnesses
    Update statistics
End while
```

Sphere Function: Definition

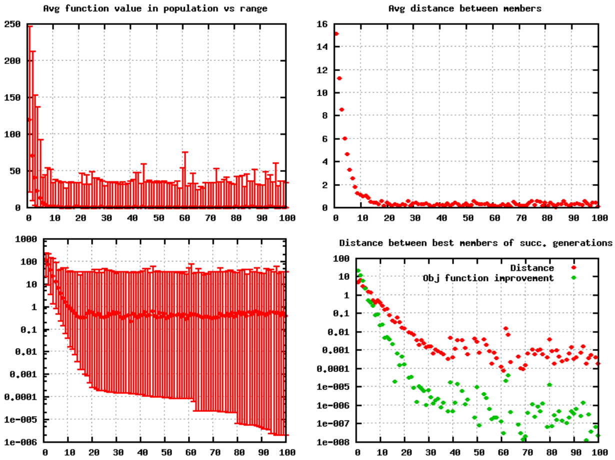
- *Definition:* $f(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$
- *Search domain:* $x_i \in [-6, 6],\ i = 1, 2, \ldots, n$
- *Number of local minima:* no local minima, only global one
- *The global minimum:* $\mathbf{x}^* = (0, \ldots, 0),\ f(\mathbf{x}^*) = 0$



Sphere Function: Algorithm Parameters

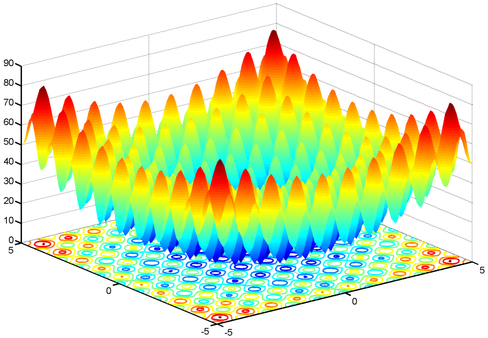
- N = 10
- Population size = 1000
- Initial population size = 0
- Reproduction params: Number of elite = 10, Mutation rate = 0.2
- Crossover params: Heuristic, Ratio = 0.8, Randomize On
- Fitness scaling: Rank
- Selection: Roulette
- Mutation params: Uniform, Gene Mutation Probability = 0.01
- Areal: $[-6.01250509, 6.01250509] \times N$, Killing On
- Max generations = 100
- **Best value** = 0.1984614290024165E-05
- **Time** = 0h 4m 2s

Sphere Function: Optimization Process



Rastrigin's Function: Definition

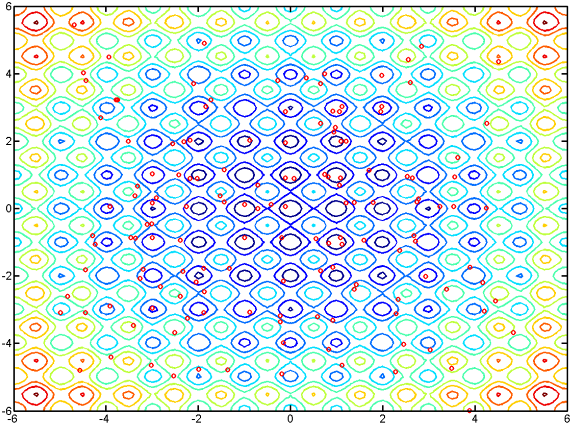
- *Definition:* $f(\mathbf{x}) = 10n + \sum_{i=1}^{n} \left(x_i^2 - 10\cos(2\pi x_i)\right)$
- *Search domain:* $x_i \in [-6, 6], \; i = 1, 2, \ldots, n$
- *Number of local minima:* several local minima
- *The global minimum:* $\mathbf{x}^* = (0, \ldots, 0), \; f(\mathbf{x}^*) = 0$



Rastrigin's Function: Algorithm Parameters

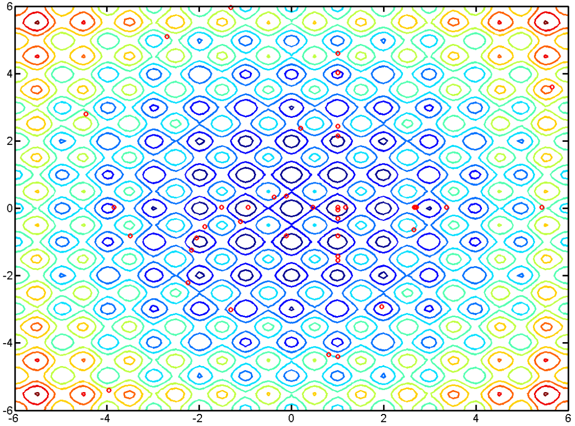
- N = 10
- Population size = 1000
- Initial population size = 0
- Reproduction params: Number of elite = 10, Mutation rate = 0.2
- Crossover params: Heuristic, Ratio = 0.8, Randomize On
- Fitness scaling: Rank
- Selection: Roulette
- Mutation params: Uniform, Gene Mutation Probability = 0.01
- Areal: $[-6.01250509, 6.01250509] \times N$, Killing On
- Max generations = 100
- **Best value** = 0.1001886961046239E-01
- **Time** = 0h 4m 43s

Rastrigin's Function, Generation = 1



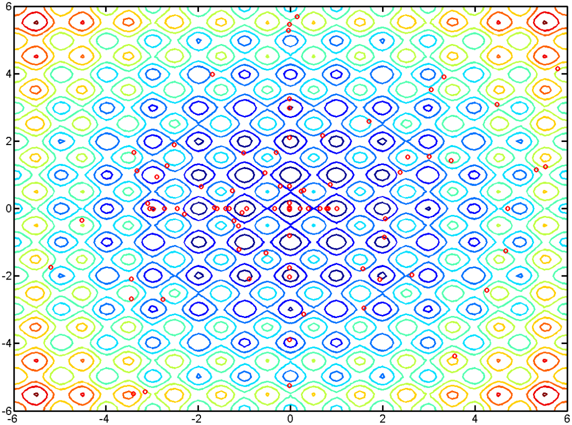
objects

Rastrigin's Function, Generation = 10



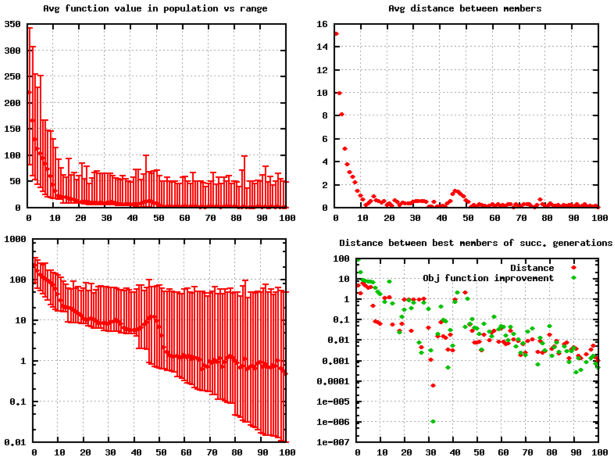
stop

Rastrigin's Function, Generation = 60



stop

Rastrigin's Function: Optimization Process

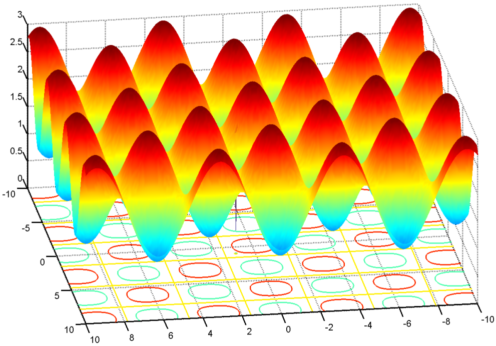


Rastrigin's Function: Different Params — Different Results

Scaling	Elite	Mutation	Crossover	Result	Time
Rank	10	Unif(0.01)	Heur(0.8, 1)	0.196	0h 4m 27s
Rank	10	**Gauss(1,1)**	Heur(0.8, 1)	3.082	0h 4m 25s
Rank	10	Unif(0.01)	Heur(0.8, **0**)	0.100E-01	0h 4m 43s
Rank	10	Unif(**0.1**)	Heur(0.8, 1)	0.593E-02	0h 4m 30s
Rank	**0**	Unif(**0.1**)	Heur(0.8, 1)	0.125E-03	0h 4m 29s
Linear	**0**	Unif(**0.1**)	Heur(0.8, 1)	7.4327	0h 4m 1s

CosExp Function: Definition

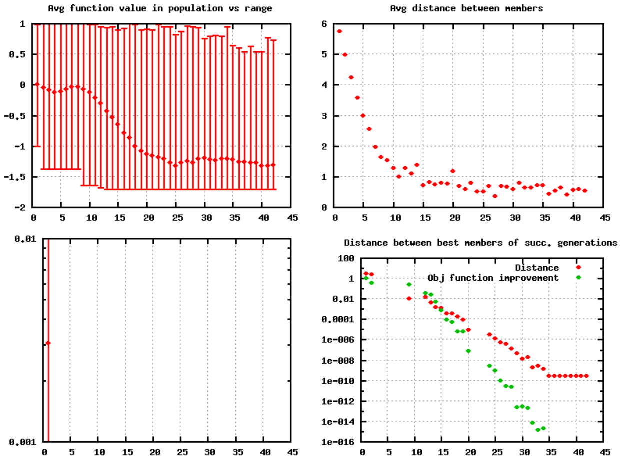
- *Definition:* $f(\mathbf{x}) = \cos(x_1)\cos(x_2) - 2 * e^{(-500((x_1-1)^2+(x_2-1)^2))}$
- *Search domain:* $x_i \in [-6, 6],\ i = 1, 2$
- *Number of local minima:* many local minima
- *The global minimum:* $\mathbf{x}^* = (1, \ldots, 1),\ f(\mathbf{x}^*) = -1.7081$



CosExp Function: Algorithm Parameters

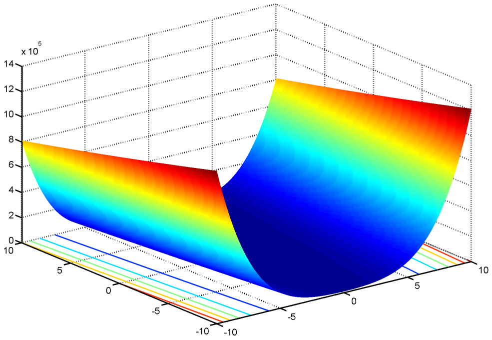
- N = 2
- Population size = 1000
- Initial population size = 0
- Reproduction params: Number of elite = 5, Mutation rate = 0.2
- Crossover params: Heuristic, Ratio = 0.8, Randomize On
- Fitness scaling: Rank
- Selection: Roulette
- Mutation params: Uniform, Gene Mutation Probability = 0.1
- Areal: $[-6.01250509, 6.01250509] \times N$, Killing On
- Max generations = 100
- **Best value** = -1.708176752160731
- **Time** = 0h 0m 49s

CosExp Function: Optimization Process



Rosenbrock's Function: Definition

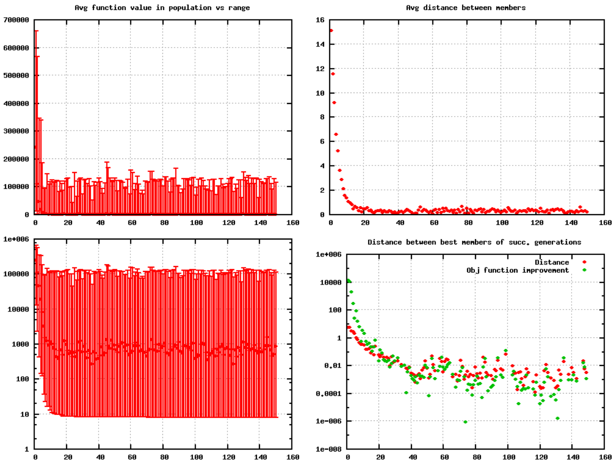
- *Definition:* $f(\mathbf{x}) = \sum_{i=1}^{n-1} \left(100\left(x_i^2 - x_{i+1}\right)^2 + (x_i - 1)^2\right)$
- *Search domain:* $x_i \in [-5, 10],\ i = 1, 2, \ldots, n$
- *Number of local minima:* several local minima
- *The global minimum:* $\mathbf{x}^* = (1, \ldots, 1),\ f(\mathbf{x}^*) = 0$



Rosenbrock's Function: Algorithm Parameters

- N = 10
- Population size = 1000
- Initial population size = 0
- Reproduction params: Number of elite = 10, Mutation rate = 0.2
- Crossover params: Heuristic, Ratio = 0.8, Randomize On
- Fitness scaling: Rank
- Selection: Roulette
- Mutation params: Uniform, Gene Mutation Probability = 0.01
- Areal: $[-6.01250509, 6.01250509] \times N$, Killing On
- Max generations = 150
- **Best value** = 7.940674306488130
- **Time** = 0h 6m 46s

Rosenbrock's Function: Optimization Process



Beam — ensemble of particles with similar coordinates

- Laboratory:

$$\mathbf{z}(t) = (x, p_x, y, p_y, z, p_z)^{\mathrm{T}}$$

- Curvilinear:

$$\mathbf{z}(s) = \begin{pmatrix} x \\ a = p_x/p_0 \\ y \\ b = p_y/p_0 \\ l = k(t - t_0) \\ \delta = (E - E_0)/E_0 \end{pmatrix}$$

$\mathbf{z}(0)$ — reference particle (often fixed point)

Equations of motion

$$\frac{d\mathbf{p}}{dt} = q\left(\mathbf{E} + \mathbf{v} \times \mathbf{B}\right)$$

Equations of motion in curvilinear coordinates:

$$\begin{aligned}
x' &= a(1+hx)\frac{p_0}{p_s} \\
y' &= b(1+hx)\frac{p_0}{p_s} \\
a' &= \left(\frac{1+\eta}{1+\eta_0}\frac{p_0}{p_s}\frac{E_x}{\chi_{e0}} + b\frac{B_z}{\chi_{m0}}\frac{p_0}{p_s} - \frac{B_y}{\chi_{m0}}\right)(1+hx) + h\frac{p_0}{p_s} \\
b' &= \left(\frac{1+\eta}{1+\eta_0}\frac{p_0}{p_s}\frac{E_y}{\chi_{e0}} + \frac{B_x}{\chi_{m0}} - a\frac{B_z}{\chi_{m0}}\frac{p_0}{p_s}\right)(1+hx) \\
l' &= \left((1+hx)\frac{1+\eta}{1+\eta_0}\frac{p_0}{p_s} - 1\right)\frac{k}{v_0} \\
\delta' &= 0
\end{aligned}$$

Map methods

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}, t),\ \mathbf{x}(0) = \mathbf{x}_\mathrm{i}$$

- Flow $\mathcal{M}_T$ establishes a mapping between the initial position $\mathbf{x}_\mathrm{i}$ at the $t = 0$ and the final position $\mathbf{x}_\mathrm{f}$ that the object assumes at the time T:

  $$\mathbf{x}_\mathrm{f} = \mathcal{M}_T(\mathbf{x}_\mathrm{i}).$$

  Especially useful for periodic systems (circular particle accelerators!)
- Map $\mathcal{M}_T$ is often hard or impossible to obtain in closed form, so it calculated via numerical integration of the equations of motion. If the function $\mathbf{f}$ is only weakly nonlinear, can use Taylor expansion.
- Differential Algebra allows to obtain it inexpensively and automatically to any order.
- Composition property: if we have have two maps: $\mathcal{M}_{t_0,t_1}$ $\mathcal{M}_{t_1,t_2}$, then

$$\mathcal{M}_{t_0,t_2} = \mathcal{M}_{t_1,t_2} \circ \mathcal{M}_{t_0,t_1} \tag{1}$$

Map methods, COSY Infinity notation

$$x_{\mathrm{f}} = (x|x)x_{\mathrm{i}} + (x|a)a_{\mathrm{i}} + (x|y)y_{\mathrm{i}} + (x|b)b_{\mathrm{i}} + (x|l)l_{\mathrm{i}} + (x|\delta)\delta_{\mathrm{i}}$$
$$+ (x|xx)x_{\mathrm{i}}^2 + (x|xa)x_{\mathrm{i}}a_{\mathrm{i}} + (x|xy)x_{\mathrm{i}}y_{\mathrm{i}} + (x|xb)x_{\mathrm{i}}b_{\mathrm{i}} + \ldots .$$

$$x_{\mathrm{f}} = \sum (x|x^{i_1}a^{i_2}y^{i_3}b^{i_4}l^{i_5}\delta^{i_6})x_{\mathrm{i}}^{i_1}a_{\mathrm{i}}^{i_2}y_{\mathrm{i}}^{i_3}b_{\mathrm{i}}^{i_4}l_{\mathrm{i}}^{i_5}\delta^{i_6}$$
$$a_{\mathrm{f}} = \sum (a|x^{i_1}a^{i_2}y^{i_3}b^{i_4}l^{i_5}\delta^{i_6})x_{\mathrm{i}}^{i_1}a_{\mathrm{i}}^{i_2}y_{\mathrm{i}}^{i_3}b_{\mathrm{i}}^{i_4}l_{\mathrm{i}}^{i_5}\delta^{i_6}$$
$$y_{\mathrm{f}} = \sum (y|x^{i_1}a^{i_2}y^{i_3}b^{i_4}l^{i_5}\delta^{i_6})x_{\mathrm{i}}^{i_1}a_{\mathrm{i}}^{i_2}y_{\mathrm{i}}^{i_3}b_{\mathrm{i}}^{i_4}l_{\mathrm{i}}^{i_5}\delta^{i_6}$$
$$b_{\mathrm{f}} = \sum (b|x^{i_1}a^{i_2}y^{i_3}b^{i_4}l^{i_5}\delta^{i_6})x_{\mathrm{i}}^{i_1}a_{\mathrm{i}}^{i_2}y_{\mathrm{i}}^{i_3}b_{\mathrm{i}}^{i_4}l_{\mathrm{i}}^{i_5}\delta^{i_6}$$
$$l_{\mathrm{f}} = \sum (l|x^{i_1}a^{i_2}y^{i_3}b^{i_4}l^{i_5}\delta^{i_6})x_{\mathrm{i}}^{i_1}a_{\mathrm{i}}^{i_2}y_{\mathrm{i}}^{i_3}b_{\mathrm{i}}^{i_4}l_{\mathrm{i}}^{i_5}\delta^{i_6}$$
$$a_{\mathrm{f}} = \sum (a|x^{i_1}a^{i_2}y^{i_3}\delta^{i_4}l^{i_5}\delta^{i_6})\delta_{\mathrm{i}}^{i_1}a_{\mathrm{i}}^{i_2}y_{\mathrm{i}}^{i_3}b_{\mathrm{i}}^{i_4}l_{\mathrm{i}}^{i_5}\delta^{i_6}$$

Problem Description

- Strong focusing by quadrupoles (magnetic lens) — main element of modern accelerators
- Linear map (matrix) — linear optics properties, combination — matrix multiplication
- Stigmatic (simultaneous) imaging, or point-to-point system, important for collider IR
- Smallest system to achieve stigmatic imaging — quadrupole triplet, demo in **demo.fox**

```
MQ .1 -q1 .025 ;
DL .06 ;
MQ .1 q2 .035 ;
DL .06 ;
MQ .1 -q1 .025 ;
```

- Map methods of COSY Infinity — arbitrary map elements access

Problem Description (cont.)

If x — position of the ray, m — its slope

$$M = \begin{pmatrix} (x,x) & (x,m) \\ (m,x) & (m,m) \end{pmatrix}$$

and

$$\begin{pmatrix} x_f \\ m_f \end{pmatrix} = \begin{pmatrix} (x,x) & (x,m) \\ (m,x) & (m,m) \end{pmatrix} \begin{pmatrix} x_i \\ m_i \end{pmatrix} \tag{2}$$

Imaging systems is an optical systems: final position of a ray is independent of its initial angle and depends only on the initial position, hence for them

$$(x,m) = 0$$

For quadrupole lens and in (x-a) and (y-b) planes stigmatic imaging condition is then:

$$(x,a) = (y,b) = 0$$

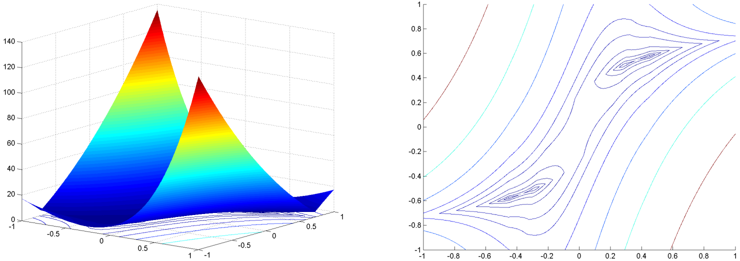
Problem Formulation

- Map is calculated by COSY Inifinity
- Objective function to be minimized is

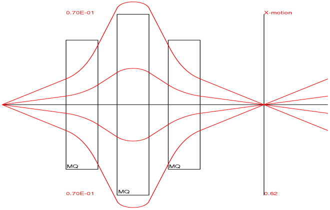
$$f(q_1, q_2) = |(x|a)| + |(y|b)| \geq 0, \ \forall q_1, q_2,$$

  and we are interested in solutions that bring it to 0.
- 4 solutions. Conventional methods require initial guesses to find all of them
  1. $q_1 \approx 0.452$, $q_2 \approx 0.58$,
  2. $q_1 \approx 0.288$, $q_2 \approx 0.504$,
  3. $q_1 \approx -0.288$, $q_2 \approx -0.504$,
  4. $q_1 \approx -0.452$, $q_2 \approx -0.58$.

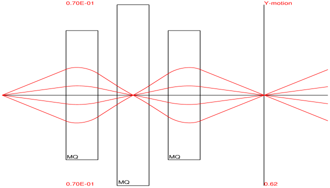
Objective Function



Solution 1

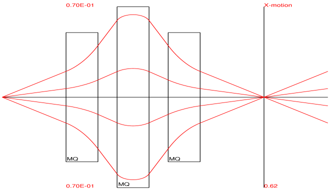


(e) (x-z) projection

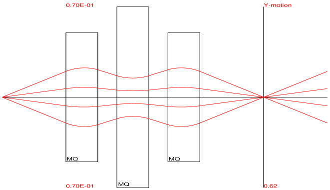


(f) (y-z) projection

Solution 2

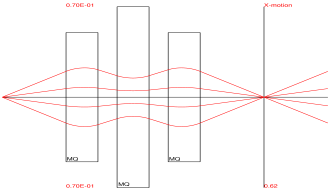


(g) (x-z) projection

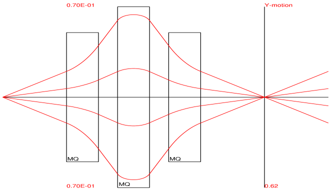


(h) (y-z) projection

Solution 3

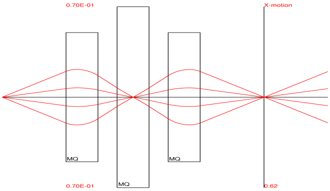


(i) (x-z) projection

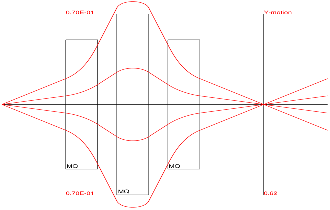


(j) (y-z) projection

Solution 3



(k) (x-z) projection



(l) (y-z) projection

Results

Search space $S = [-10, 10] \times [-10, 10]$, population size = 100*dimension = 200

# Runs	Solution found (%)			
	1	2	3	4
200	12.0	46.5	36.0	5.5
1000	9.0	46.9	37.0	7.1
3000	4.7	31.3	60.3	3.7
10000	8.18	47.27	38.19	6.36

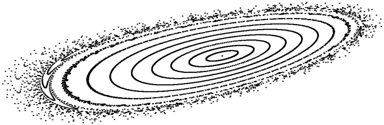
Conclusions

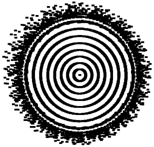
- GATool was able to find one solution every run, all solutions were found at least once on 200 runs
- GATool was able to find really sharp minima with almost no human intervention (human time is expensive!)
- Established method is not limited to linear map elements and simple structures

Problem Description

Normal Form Defect Function is a tool for rigorous studies of the circular accelerator stability. In Normal Form coordinates particles follow almost perfect circles around a fixed point. NFDF measures this non-perfection (I — invariants of motion):

$$d = \max(I(\mathcal{M}) - I)$$

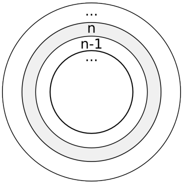


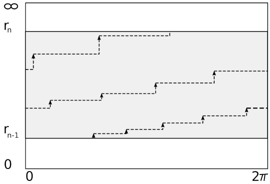


Phase space trajectories in FODO cell, obtained for 1000 turns by applying one turn map to the vector with initial coordinates 1000 times; in conventional (left) and normal form (right) coordinates

Problem Description (cont.)

- Rigorous estimations of the stability ranges for perturbed motion exist, but allow predictions of stability only for very small perturbations and are totally dominated by realistic construction errors.
- Can estimate stability for a finite, but still practically useful, time, applying principles established by Nekhoroshev
- Divide the normal form coordinate space for each degree of freedom into a set of rings such that in each of them motion is almost circular





Problem Description (cont.)

If for the ring n the defect is not larger than Δr_n, then all particles launched from ring $(n-1)$ need to make at least

$$N_n = \frac{r_n - r_{n-1}}{\Delta r_n}$$

turns before they reach the n-th ring. Then min number of turns inner circle to get from $r_{\min}$ (initial region) to the $r_{\max}$

$$r_{\min} = r_1 < r_2 < \cdots < r_n = r_{\max}.$$

If maximal defects on each of the rings Δr_i, $i = \overline{2, n}$

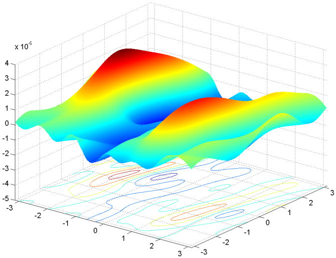
$$N = \sum_{i=2}^{n} \frac{r_i - r_{i-1}}{\Delta r_i}.$$

Usually Δr_i are small $\Rightarrow$ motion stability can be assured for a large number of turns.

Motivation

- Need tight and rigorous bounds for Δr_i, served as a motivation for COSY-GO. One more COSY-GO + GATool hybridization test
- In practice, NFDFs are multi-dimensional multi-modal polynomials of high order, with many of the high-order elements cancel each other, behaviour of those functions is highly oscillatory and they quickly grow with radii. Thus they pose substantial difficulties for conventional optimization methods. Good test functions

Synthetic, 5th order, 6-dimenstional




```
[ 0.499999999E-001, 0.100000001 ]  [ -3.14159266,  3.14159266 ]
[ 0.499999999E-001, 0.100000001 ]  [ -3.14159266,  3.14159266 ]
[ 0.499999999E-001, 0.100000001 ]  [ -3.14159266,  3.14159266 ]
```

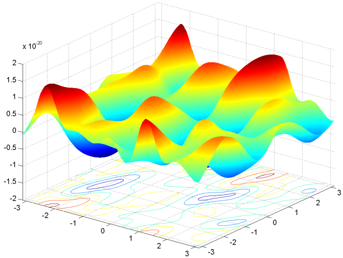
Range estimate: $0.5 \cdot 10^{-4}$

Results

Method	Time (s)	Max Value	Difference with TM method
TM method	256 x 3297[1]	-	[-, -]
Naive Sampling	109	0.209075292E-4	[1.28294580E-5, 1.28294626E-5]
GATool, pop=60	17	0.327416142E-4	[9.95373092E-7, 9.95377677E-7]
GATool, pop=180	83	0.319524687E-4	[1.78451855E-6, 1.78452314E-6]
GATool, pop=300	300	0.332044502E-4	[5.32537049E-7, 5.32541634E-7]
GATool, pop=600	378	0.331694477E-4	[5.67539577E-7, 5.67544162E-7]
GATool, pop=1000	553	0.332085478E-4	[5.28439469E-7, 5.28444054E-7]
GATool, pop=1200	613	0.336515785E-4	[8.54087318E-8, 8.54133164E-8]
GATool, pop=2000	3459	0.337010630E-4	[3.59242826E-8, 3.59288671E-8]

[1] Wall clock time

Realistic, 7th order, 4-dimenstional (Tevatron map, courtesy of P.Snopok)




```
[ 0.199999999E-004, 0.400000001E-004 ]  [ -3.14159266,  3.14159266 ]
[ 0.199999999E-004, 0.400000001E-004 ]  [ -3.14159266,  3.14159266 ]
```

Results

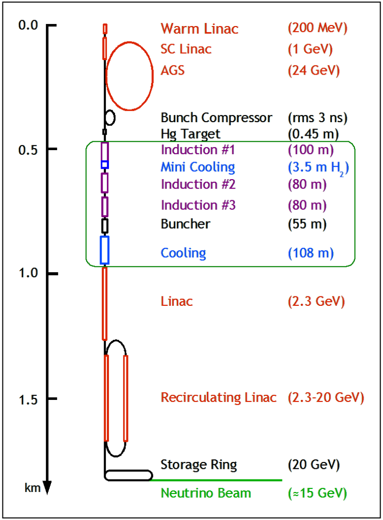
Method	Time (s)	Max Value	Difference with TM method
TM method	1024 x 935[2]	-	[-, -]
Naive Sampling	46	0.384215054E-18	[4.01596187E-22, 7.11441777E-14]
GATool, pop=40	5	0.380347985E-18	[4.26866555E-21, 7.11441816E-14]
GATool, pop=200	18	0.382665745E-18	[1.95090547E-21, 7.11441793E-14]
GATool, pop=400	75	0.384126132E-18	[4.90518103E-22, 7.11441778E-14]
GATool, pop=600	177	0.384406960E-18	[2.09690285E-22, 7.11441775E-14]
GATool, pop=800	117	0.384035970E-18	[5.80680790E-22, 7.11441779E-14]
GATool, pop=1000	230	0.384644775E-18	[-2.81241401E-23, 7.11441773E-14]

[2] Wall clock time

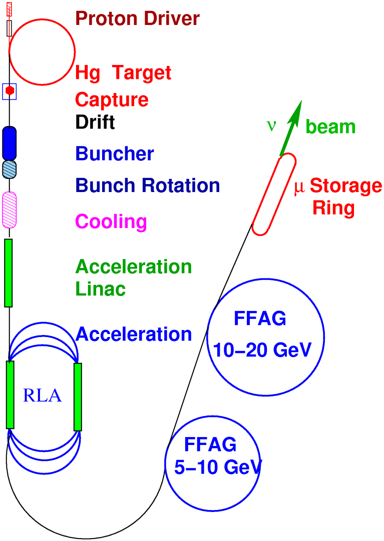
Conclusions

- GATool is fast and efficient enough to be used for cutoff updates with COSY-GO
- GATool is efficient on “nasty” functions

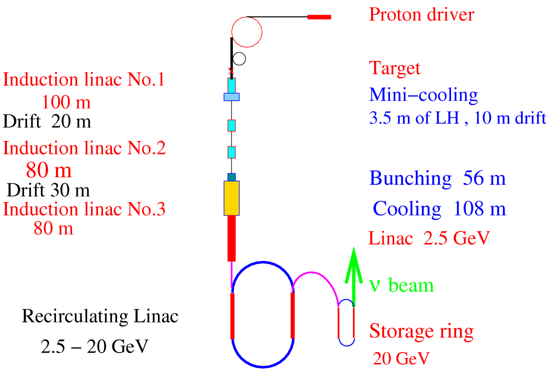
Muon Accelerators: Neutrino Factory (oder design)



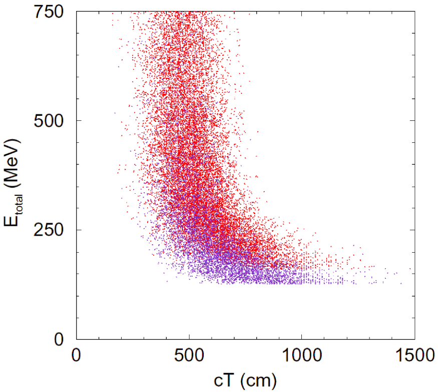
Muon Accelerators: Neutrino Factory (Study 2a)



Muon Accelerators: Muon Collider



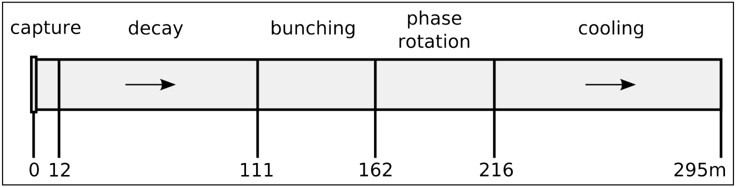
Initial Beam



Distribution of particles energies 12m from the target calculated by MARS, $E_{\text{total}} = E_0 + T$, where E_0 is a rest energy (105.6 MeV for muons), T — kinetic energy

Front End

The baseline Front End schematics from the latest International Scoping Study



Problem

- Control parameters (fields, positions, materials, etc)
- R&D: find optimal parameters as to satisfy requirements on:
  - Capture: Matching Emittance (phase space volume) to Acceptance
  - Maximize production: muons survived and captured
  - Minimize cost (length, fields, frequencies,...)
  - Set of optimal designs to choose from
  - ...

Methodology

- Number of survived particles within acceptance — objective function
- COSYInfinity + GATool — optimizer (population size = 250)
- ICOOL + ECALC9 — simulation code (2000 particles, 0.4hrs, PIV) and production analysis
- Perl — driver that controls execution and “glues” programs together
- Short version of Front End design from Neuffer, cooling section for optimization
- Varied control parameters:
  - RF frequency in cooling section (also influences the following accelerator section): $\nu_{\mathrm{rf,cool}} \in [200, 204]$ MHz.
  - RF field gradient in cooling section: $V_{\mathrm{rf,cool}} \in [12, 20]$ MV/m.
  - RF field phase in cooling section: $\varphi_{\mathrm{rf,cool}} \in [0, 360]$ degrees.
  - Central momentum in the first 4 matching sections of the cooling channel: $p_{\mathrm{c,match\_cool}} \in [0.22, 0.24]$ GeV/c.

Methodology (cont.)

- Acceptance estimate:
  - minimum and maximum p_z: 0.100 GeV/c and 0.300 GeV/c, correspondingly;
  - transverse acceptance cut: 30E-3 m·rad;
  - longitudinal acceptance cut: 0.25 m·rad;
  - RF frequency for the bucket calculation set to a value used by RF cavities of the cooling section (on of the control parameters).

Results

Parameters	$\nu_{\mathrm{rf,cool}}$ [MHz]	$V_{\mathrm{rf,cool}}$ [MV/m]	$\varphi_{\mathrm{rf,cool}}$ [degrees]	$p_{\mathrm{c,match\_cool}}$ [GeV/c]	n_2 (n_2/2000) particles	n_2 (n_2/8000) particles
reference parameters	201.25	18.00	30.000	0.220	498 (0.249)	1740 (0.218)
3rd opt. run, 6th best	201.46	17.77	11.320	0.229	480 (0.240)	1791 (0.224)
3rd opt. run, best	201.40	17.06	12.648	0.226	492 (0.246)	1782 (0.223)
1st opt. run, best	200.55	17.10	26.970	0.220	467 (0.234)	1780 (0.222)
3rd opt. run, 5th best	201.28	17.76	12.457	0.226	484 (0.242)	1773 (0.222)
3rd opt. run, 3rd best	201.47	17.67	13.470	0.228	485 (0.243)	1762 (0.220)
3rd opt. run, 2nd best	201.42	17.68	12.555	0.226	486 (0.243)	1750 (0.219)
3rd opt. run, 7th best	201.34	17.68	12.020	0.226	479 (0.240)	1746 (0.218)
2nd opt. run, 2nd best	201.24	18.91	20.520	0.228	471 (0.236)	1714 (0.214)
3rd opt. run, 4th best	201.48	17.75	11.860	0.227	485 (0.243)	1669 (0.209)
2nd opt. run, best	201.20	18.88	22.477	0.230	497 (0.249)	1643 (0.205)

COSY Infinity

- Scientific computing code based on Differential Algebra (DA) and Taylor Model (TM) methods
- Primary applications: Beam Theory, Accelerator Design, Rigorous Computing, Rigorous Integration and Optimization, high-order Automatic Differentiation,
- Features: arbitrary order for maps of the dynamical systems, parameter-dependent maps, no approximations in motion or field description, Normal Form methods, fast fringe field models extensive library of standard elements, flexible input language (COSYScript) with built-in optimization syntax and graphics output
- Large user base: > 1000 as of 2004

Available at `www.cosyinfinity.org`

COSY++

- New file inclusion mechanism for increased modularity
- Macroprogramming with Perl from COSYScript via Active Blocks
- Enhanced command-line interface
- GATool
- Library of convenience functions including vector operations similar to MatLab
- Automatic conversion of the old-syntax scripts

Summary

- GATool framework for the continuous optimization of the real-valued functions is implemented in COSY Infinity and tested
- GATool application on various Accelerator Design problems is studied, its usefulness is verified on test and real-life problems; potentially more applications
- Neutrino Factory Front End optimization is performed, practically useful results obtained, general framework for the Front End optimization is suggested, implemented and tested

Representation and Fitness Scaling

- **Representation:** vectors of real numbers from search domain

$$S = [a_1, b_1] \times [a_2, b_2] \times \ldots \times [a_v, b_v]$$

- **Fitness scaling:** ($f \to \text{fitness} > 0$)
  - *Linear:*

$$\text{fitness}(\mathbf{x}_i) = \text{fitness}_i = \overline{f} - f_i \geq 0$$

  - *Proportional:*

$$\text{fitness}_i = \begin{cases} \left(\frac{\overline{f}+\underline{f}}{2} - f_i \right) & \text{if } \underline{f} \geq 0 \\ \left(\frac{\overline{f}+\underline{f}}{2} - f_i \right) + \underline{f} & \text{if } \underline{f} < 0 \end{cases}$$

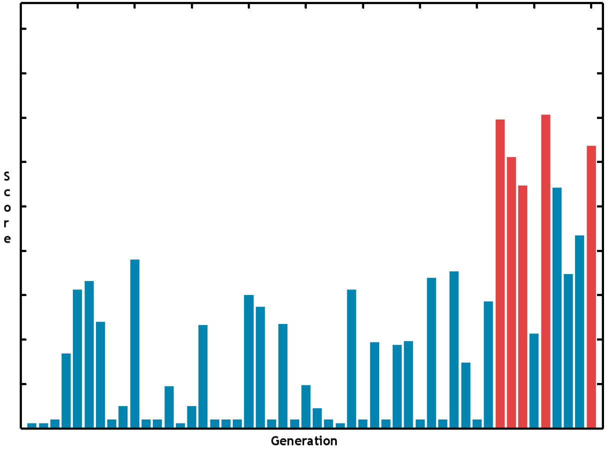
  - *Rank:* sort in ascending order, then

$$\text{fitness} = \frac{1}{\sqrt{i}}$$

Evolutionary Operators and Selection

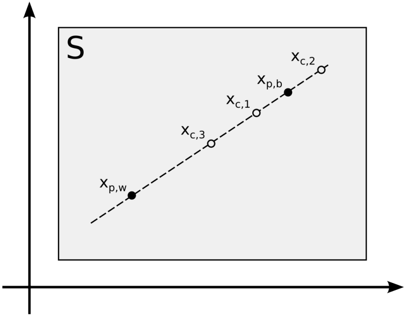
- **Genetic operators:**
  - Elitism (preservation), number of elite
  - Uniform mutation, Gaussian mutation (exploration), mutation rate
  - Continuous crossover (exploitation)
- **Selection:**
  - Stochastic Uniform
  - Roulette Wheel
  - Tournament

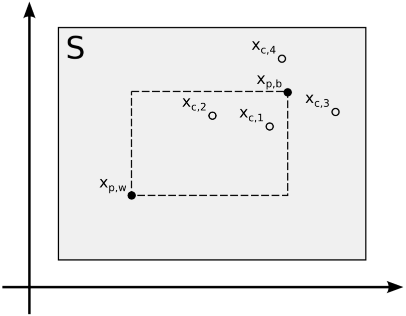
Elitism



Evolutionary Operators: Crossover

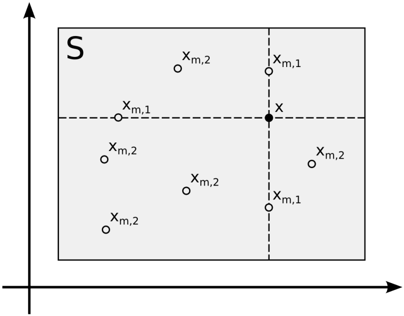
$$\mathbf{x}_{\mathrm{c}} = \mathbf{x}_{\mathrm{p,w}} + \beta(\mathbf{x}_{\mathrm{p,b}} - \mathbf{x}_{\mathrm{p,w}})$$





Evolutionary Operators: Uniform Mutation

p_c — mutation rate, $x_{j,\mathrm{m}} = \mathrm{rand}[a_j, b_j]$



Evolutionary Operators:Gaussian Mutation

$$\mathbf{x}_{\mathrm{m}} = \mathbf{x} + \Delta\mathbf{x}$$

$$\Delta x_j \sim N(\mu, \sigma^2) = N\left(0, \frac{b_j - a_j}{2}\right)$$

if adaptive:

$$\sigma^2 = \sigma^2(g) = \left(1 - \alpha \frac{g}{g_{\mathrm{max}}}\right)$$

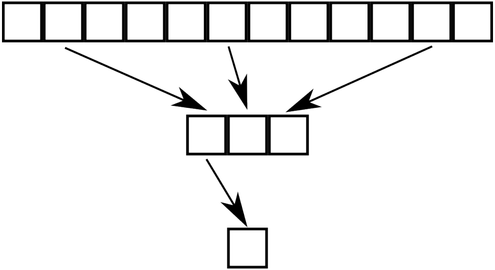
Selection: Stochastic Uniform and Tournament

Stochastic Uniform

$$S_k = \sum_{i=1}^{k} \text{fitness}_i,\ k = \overline{1, N},\ h = \frac{S_N}{N_{\text{select}}},\ t_1 = \text{rand}[0, h]$$

$$t_j = t_1 + (j-1)h,\ j = \overline{2, N_{\text{select}}}.$$

Tournament



Selection: Roulette Wheel

Statistics and Stopping Criteria

Diversity!

- **Statistics:**
  - Objective function values range: $\Delta f = \overline{f} - \underline{f}$
  - Average function value over population
  - Average distance between population members (estimated, sampling: 5-10%)
  - Improvements from generation to generation
- **Stopping criteria:**
  - Maximum number of generations
  - Maximum number of stall generations + tolerance
  - Desired objective function value
  - Time limit

Types of Noise

- *Static:* the function values contain errors but those errors remain the same every time the function is evaluated:

$$f(x) = f_{\text{true}}(x) + \Delta f(x), \ \forall x.$$

- *Dynamic:* the function values contain errors that change every time the function is evaluated:

$$f(x) = f_{\text{true}}(x) + \text{rand}(-\Delta f(x), +\Delta f(x)),$$

where rand is a random number whose distribution is specified by the considered problem. For simplicity here we consider only uniformly distributed random numbers.

Static Noise Example

Test problems: main function + noise function

Sphere function:

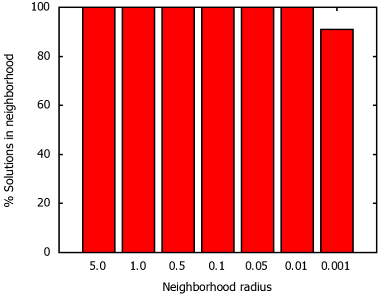
$$f(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$$

Rastrigin function:

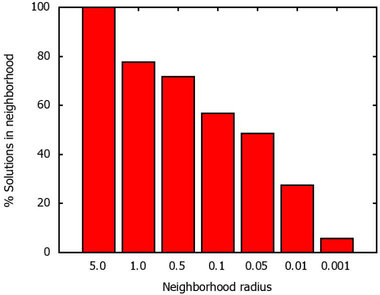
$$f(\mathbf{x}) = 10n + \sum_{i=1}^{n} \left(x_i^2 - 10\cos(2\pi x_i)\right).$$

Same global minimum: $\mathbf{x}^* = \mathbf{0}$, $f(\mathbf{0}) = 0$

GATool Results, Population = 10*dim = 50, 100 Runs



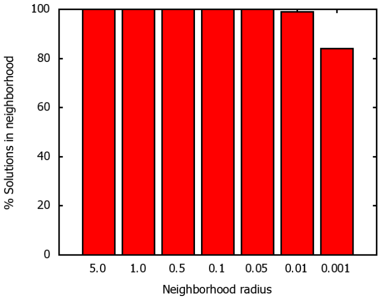
(o) Sphere, avg. time = 4.09 sec



(p) Rastrigin, avg. time = 5.22 sec

GATool Results, Population = 20*dim = 100, 100 Runs

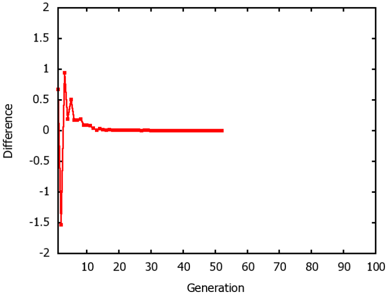
Increase the population size!



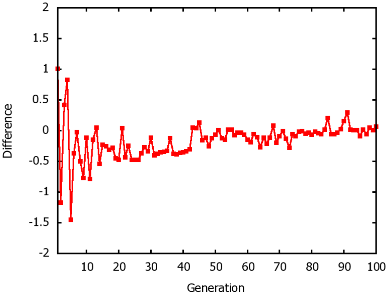
Rastrigin, avg. time = 11.92 sec

Dynamic Noise Example

Elitism does not work!



(q) No noise



(r) Dynamic noise

Figure: GATool, 5-dim Sphere, population size 50. Generation number versus $\sum_{i=1}^{v}(x_i^* - x_{i,\mathrm{true}})$, where $\mathbf{x}^*$ — the best minimizer found by GATool, $\mathbf{x}_{\mathrm{true}}$ — the true global minimizer (in this case $\mathbf{0}$), noise from the $[-1, 1]$ range

Averaging Strategy

$$\mathbf{x}^* = \overline{\mathbf{x}^*} = \frac{1}{g_2 - g_1 + 1} \sum_{i=g_1}^{g_2} \mathbf{x}_i^*, \; 1 \le g_1 \le g_2 \le g_{\max}.$$

typically $g_1 = 5 \ldots 20$

generation	Euclidean distance to minimizer	
	current	averaged
100	0.18567	0.22973
200	0.17075	0.31166
500	0.13479	0.07508
1000	0.21228	0.06281

COSY-GO Rigorous Global Optimization Package: Principles

- Stack of boxes, branch-and-bound method
- *Taylor Model Methods:* if $f \in C^{n+1}(D)$ then P — Taylor polynomial at $x_0 \in D$ up to order n and I — remainder error bound interval, then Taylor Model of the order n:

$$f(x) \in P(x, x_0) + I, \ \forall x \in D.$$

- *Naive Bounding:* evaluate P in interval arithmetic, add I
- *Linear Dominated Bounder (LDB):* linear part dominates, bound linear part, use to reduce domain
- *Quadratic Fast Bounder (QFB):* in the neighborhood of the minimum, Hessian is positive definite

$$P + I = (P - Q) + I + Q \Longrightarrow l(P + I) = l(P - Q) + l(I) + l(Q)$$

If we now choose Q such that $Q = Q_{x_0} = \frac{1}{2}(x - x_0)^{\mathrm{T}} H (x - x_0) \geq 0$, then $l(Q) = 0$. If we choose $\mathbf{x_0}$ to be a minimum of P_2, then lower bound is dominated by orders ≥ 3.

COSY-GO Rigorous Global Optimization Package: Algorithm (step 1)

1. A lower bound is obtained by applying the various available bounding schemes sequentially in the order described below. If the obtained lower bound is below the cutoff value, the box is eliminated, otherwise it is bisected. Each subsequent method is applied only if the previous one fails. The following bounding methods are used:

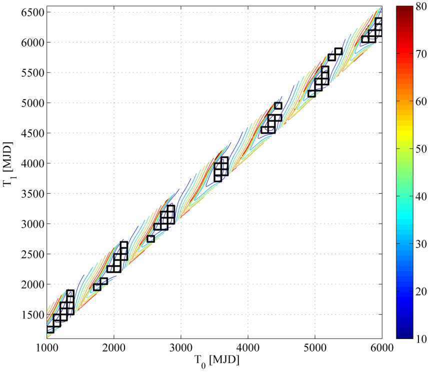
   a) Simple interval bounding of the function f.
   b) Naive Taylor model bounding based on the evaluation of the Taylor polynomial P in interval arithmetic.
   c) LDB bounding. If fails, the LDB domain reduction is performed.
   d) QFB bounding, if the quadratic part of the P is positive definite.

COSY-GO Rigorous Global Optimization Package: Algorithm (step 2)

2. The cutoff value is heuristically updated using following methods:
   a) The result of the function evaluation at the midpoint of the current box.
   b) The linear and quadratic parts of P are utilized to obtain a potential cutoff update.

better cutoff $\Longrightarrow$ more boxes eliminated $\Longrightarrow$ cheaper/faster method

Example of the Rigorous Global Optimization

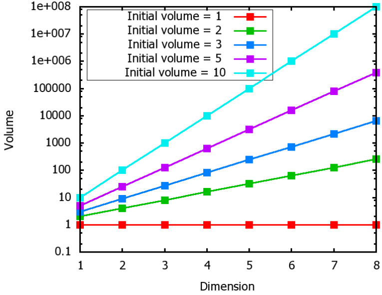


Global optimization of the spacecraft trajectories: pruned search space in the epoch/epoch plane (courtesy of Roberto Armellin)

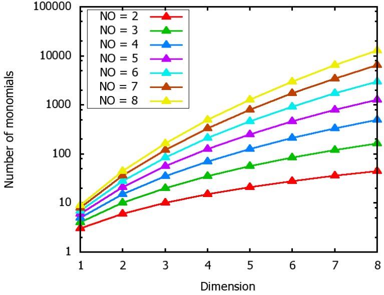
Problems of the Global Optimization with COSY-GO

$$V = d^v$$

$$M = \frac{(n+v)!}{n!v!}$$



(a) Search space volume for different initial volumes



(b) Number of monomials for different expansion orders

COSY-GO Performance for Different Dimensions

Problem		Dimension				
		2	3	5	7	9
Paviani, NO = 8	V	6.30e+1	5.11e+2	3.27e+4	2.09e+6	1.33e+8
	t	0.04	0.19	7.43	290.17	13524.51
CosExp, NO = 5	V	6.40e+1	5.12e+2	3.27e+4	2.09e+6	1.34e+8
	t	0.03	0.08	1.19	24.6	337.31
SinSin, NO = 8	V	4.00	8.00	3.20e+1	1.28e+2	5.12e+2
	t	0.17	1.37	395.53	7677.42	-.-
An, NO = 2	V	2.50e-1	1.25e-1	3.13e-2	7.81e-2	1.95e-3
	t	0.02	0.04	0.05	0.03	0.04

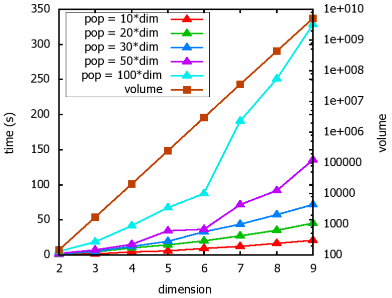
GATool Performance for Different Dimensions, Population = dim*100

Problem		Dimension				
		2	3	5	7	9
Paviani	V	6.30e+1	5.11e+2	3.27e+4	2.09e+6	1.33e+8
	t	34.25	114.64	366.69	750.87	1301.53
	Q	3.69e-6	1.89e-6	4.04e-5	8.37e-5	1.32e-3
CosExp	V	6.40e+1	5.12e+	3.27e+4	2.09e+6	1.34e+8
	t	29.15	78.72	302.23	571.32	2123.57
	Q	3.99e-15	1.92e-10	9.54e-1	9.86e-	9.96e-1
SinSin	V	4.00	8.00	3.20e+1	1.28e+2	5.12e+2
	t	16.31	12.86	135.68	385.06	685.15
	Q	0.00	4.66e-7	3.32e-7	1.91e-6	2.16e-6
An	V	2.50e-1	1.25e-1	3.13e-2	7.81e-2	1.95e-3
	t	11.14	27.01	239.82	454.95	822.86
	Q	1.11e-16	6.87e-5	9.47e-5	1.11e-3	1.85e-3

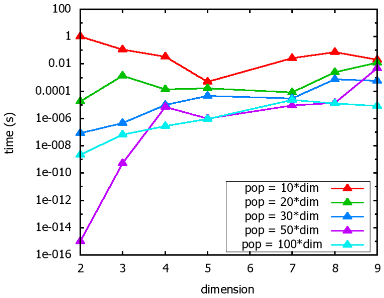
GATool Performance for Different Dimensions, Population = dim*10

Problem		Dimension				
		2	3	5	7	9
Paviani	V	6.30e+1	5.11e+2	3.27e+4	2.09e+6	1.33e+8
	t	0.89	2.33	8.48	16.10	27.24
	Q	1.16e-2	2.45e-2	1.68e-2	1.78e-1	3.02e-2
CosExp	V	6.40e+1	5.12e+2	3.27e+4	2.09e+6	1.34e+8
	t	0.86	1.52	4.44	12.13	26.12
	Q	7.13e-1	8.50e-1	9.54e-1	9.86e-1	9.96e-1
SinSin	V	4.00	8.00	3.20e+1	1.28e+2	5.12e+2
	t	0.27	1.38	4.40	15.36	33.83
	Q	3.62e-2	9.13e-3	5.12e-3	9.91-4	2.82-4
An	V	2.50e-1	1.25e-1	3.13e-2	7.81e-2	1.95e-3
	t	0.49	0.83	10.44	21.25	44.62
	Q	1.16e-3	1.27e-2	1.79e-3	9.25e-4	2.50-4

GATool Time of Execution and Quality Scaling, Different Population Sizes



(c) Time scaling



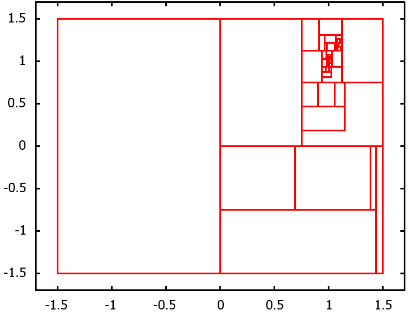
(d) Result quality scaling

Rastrigin’s function, one random run

COSY-GO + GATool Interaction Mechanism

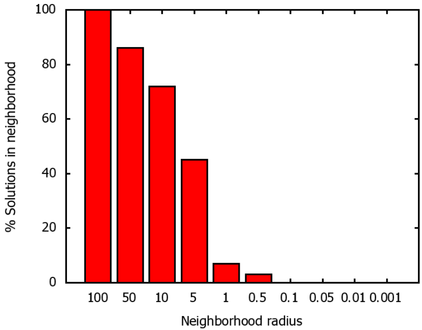
```
> GATool searches in a box and returns cutoff update
> COSY-GO uses cutoff value, performs non-rigorous and rigorous box elimination
> GATool is restarted using updated information about the search domain
  and returns new, better cutoff update.
> COSY-GO uses cutoff value, performs non-rigorous and rigorous box elimination
...
```

Boxes Considered During COSY-GO Rigorous Minimization of the 2-dimensional Rosenbrock's Function



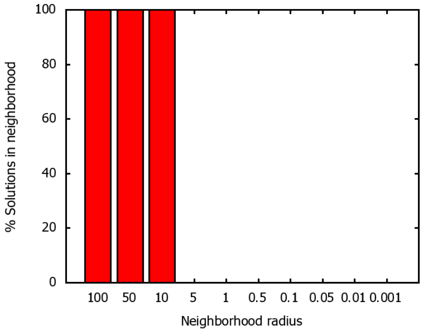
100 runs, 10-dimensional Rosenbrock's function

GATool Performance, $[-5, 10]^{10}$, $V = 5.67 \cdot 10^{11}$



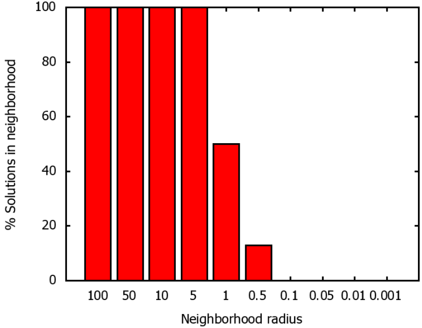
100 runs, 10-dimensional Rosenbrock's function

GATool Performance, $[-1.5, 1.5]^{10}$, $V = 5.9 \cdot 10^4$



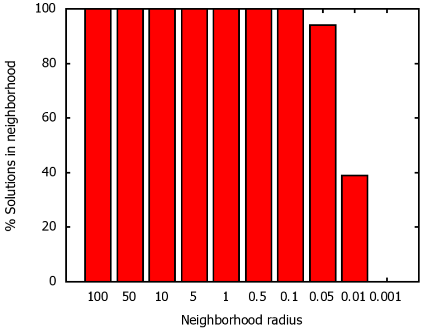
100 runs, 10-dimensional Rosenbrock's function

GATool Performance, $[0, 1.5]^{10}$, $V = 5.76 \cdot 10^1$



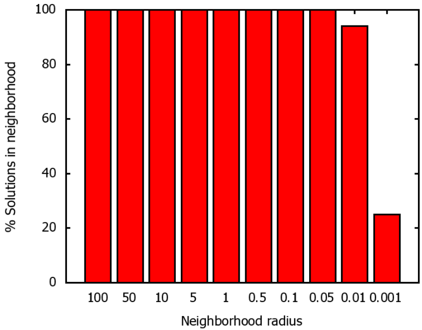
100 runs, 10-dimensional Rosenbrock's function

GATool Performance, $[0.5, 1.5]^{10}$, $V = 1.0 \cdot 10^0$



100 runs, 10-dimensional Rosenbrock's function

GATool Performance, $[0.7, 1.3]^{10}$, $V = 0.6 \cdot 10^{-2}$



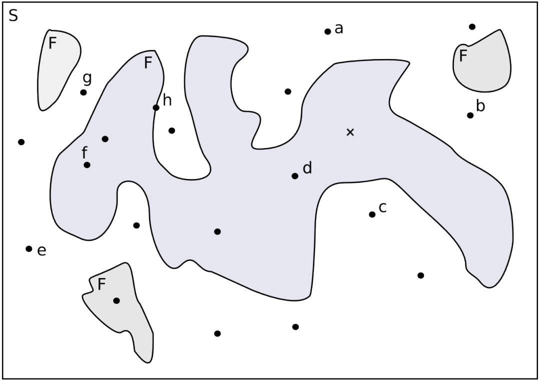
100 runs, 10-dimensional Rosenbrock's function

Conclusions

- GATool is designed and implemented as a hybrid of the best features of existing EAs
- Performance is assessed on the test problems (later on real-life problems from Accelerator Physics)
- Noise handling strategies are suggested and tested
- COSY-GO rigorous global optimizer interaction scheme is suggested, ground of the proposition is studied by experiments
  - Dependence of the computational time and quality is studied and compared to the one of COSY-GO
  - Consistency of the results, i.e. robustness of the methods is demonstrated on examples
  - Increase of the result quality with the domain reduction is demonstrated on example
- Future research:
  - Implementation of the hybrid algorithm and testing
  - More tests

Challenges

- Not originally designed to handle constraints: for unconstrained optimization fitness, for constrained — ?
- Keep or eliminate unfeasible members?
- If keep, how to compare feasible and unfeasible?



Evolutionary Algorithm (repeated)

```
Generate initial population, evaluate fitness
While stop condition not satisfied do
    Produce next population by
        Selection
        Recombination
    Evaluate fitness
End while
```

Methods

- Killing (reproduction)
- Penalty Functions (fitness evaluation)
- Special Genetic Operators (recombination)
- Selection (selection)
- Repairing (reproduction)
- Other methods (combined, one-by-one satisfaction, homomorphous mapping, co-evolution, Immune System simulation)

Penalty Function Methods Idea

Replace constrained minimization problem with unconstrained minimization problem with augmented objective function(s) so that its unconstrained minimum is the same as constrained minimum of the original problem

Penalty functions: $P_j(h_j(\mathbf{x})),\ j = \overline{1, n}$
Unconstrained multi-objective minimization problem:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in S} \mathbf{G}(\mathbf{x}),$$

where $\mathbf{G}(\mathbf{x}) = \big(P_1(h_1(\mathbf{x})), P_2(h_2(\mathbf{x})), \ldots, P_n(h_n(\mathbf{x})), f(\mathbf{x})\big)^{\mathrm{T}}$ Unconstrained single-objective minimization problem

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in S} \varphi(\mathbf{x}),$$

where $\varphi = \varphi\big(\mathbf{G}(\mathbf{x})\big)$ is the function that combines the original objective function and penalty functions into a single objective function ($\|\varphi(\mathbf{x}) - f(\mathbf{x})\| \longrightarrow 0$ as $\mathbf{x} \to F$)

Penalty Functions

- Exterior (barrier functions): $P_j(z) = -\frac{1}{h_j(\mathbf{x})}$
- Interior (power penalties): $P_j^a(h_j(\mathbf{x})) = (\max\{0, h_j(\mathbf{x})\})^a$

Combining function:

$$\varphi(\mathbf{x}) = f(\mathbf{x}) + \sum_{j=1}^{n} w_j P_j(h_j(\mathbf{x})).$$

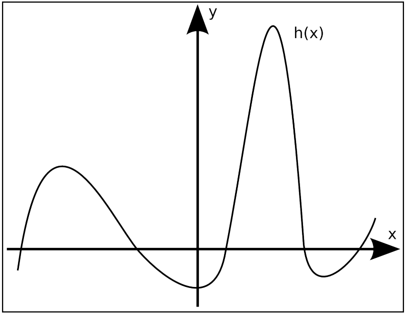
SUMT method:

$$\varphi(\mathbf{x}) = f(\mathbf{x}) - r \sum_{j=1}^{n} \frac{1}{h_j(\mathbf{x})}$$

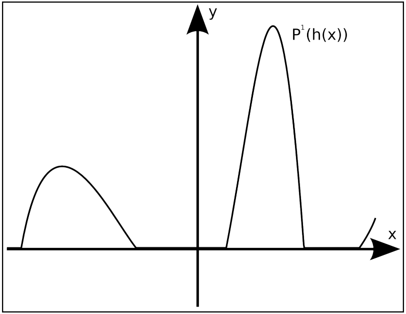
Any unconstrained minimization method (frequently used combination, $a = 2$):

$$\varphi(\mathbf{x}) = f(\mathbf{x}) + \sum_{j=1}^{n} (\max\{0, h_j(\mathbf{x})\})^2$$

Exterior Penalty Function Example



(e) Inequality constraint function



(f) Power penalty for inequality constraint function

Penalty vs Distance

$$F = \{\mathbf{x} \mid \|\mathbf{x}\| \leq 1\}, \mathbf{x} \in [-5,5]^2$$

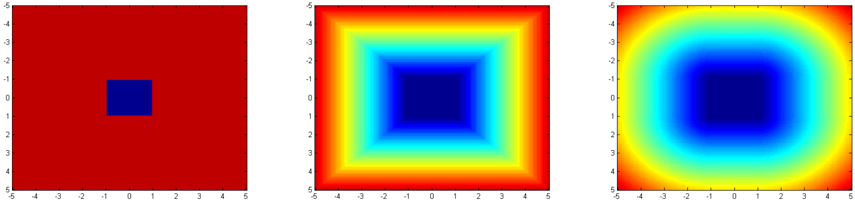


Figure: (left to right) P_0, P_1, d($\mathbf{x}$,F)

Exterior Penalty Function Types for EAs

- *Levels of Violation*

$$\varphi(\mathbf{x}) = f(\mathbf{x}) + \sum_{j=1}^{n} R_j\big(h_j(\mathbf{x})\big) P^2(h_j(\mathbf{x}))$$

- *Multiplicative*

$$\varphi = f(\mathbf{x}) P(\mathbf{x})$$

- *Dynamic*

$$\varphi(\mathbf{x}) = f(\mathbf{x}) + (Ck)^{\alpha} \sum_{j=1}^{n} P^{\beta}(h_j(\mathbf{x}))$$

$$\varphi(\mathbf{x}) = f(\mathbf{x}) + \frac{1}{2\tau_k} \sum_{j \in A} P^2(h_j(\mathbf{x}))$$

- *Adaptive*

Motivation

- Cutoff updates for COSY-GO constrained rigorous global optimization
  $\Rightarrow$ strong need for feasible points
- Problems with very expensive objective functions but much less expensive constraint functions, constraints MUST be satisfied (physical limitations)

REPA Agorithm

```
If combined penalty > penalty tolerance
    If N(0,1) < percent repaired
        If succeeded x = REFIND(x_u)
            Repair succeeded, replace x_u in population with x
        Else
            If succeeded x = REPROPT(x_u)
                Repair succeeded, replace x_u in population with x
            Else
                Repair failed
            End if
        End if
    Else
        Repair skipped
    End if
Else
    Repair not needed
End if
```

REFIND Agorithm

Find feasible individuals from the current population

$R = \{\mathbf{x}_{f,1}, \mathbf{x}_{f,2}, \ldots, \mathbf{x}_{f,N}\}$

If at least one feasible individual is found

Find $\mathbf{x}_f \in R$ such that $d(\mathbf{x}_f, \mathbf{x}_u) = \min_{\mathbf{x} \in R} d(\mathbf{x}, \mathbf{x}_u)$

Search for a feasible point along the line connecting $\mathbf{x}_u$ and $\mathbf{x}_f$ by solving

optimization problem $\lambda^* = \arg\min_\lambda P(\mathbf{x}_u(1-\lambda) + \lambda \mathbf{x}_f)$,

P --- penalty function

If resulting penalty is within tolerance

Repair succeeded, return $\mathbf{x} = \mathbf{x}_u(1-\lambda^*) + \lambda^* \mathbf{x}_f$

Else

Repair failed

End if

Else

Repair failed

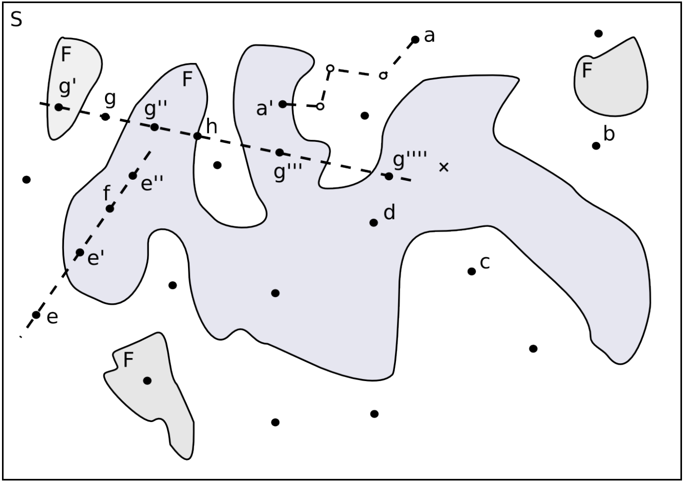
End if

REPROPT Algorithm

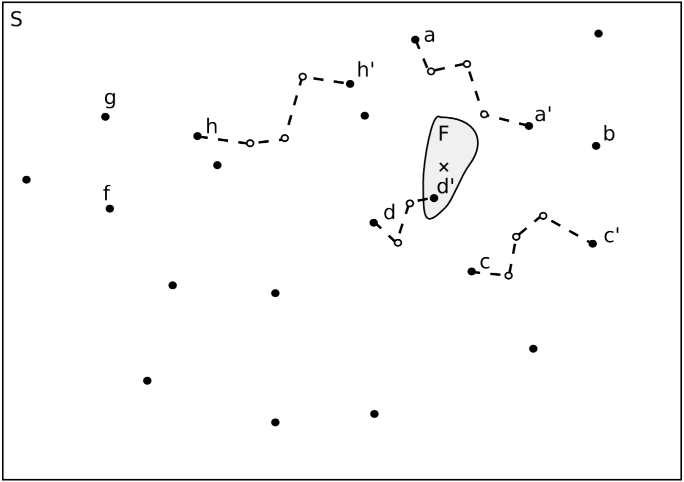
Same as REFIND... however

- there are no feasible members in the population!
- all coordinates are parameters for projection (multi-dimensional problem) $\Rightarrow$ increased complexity
- can do quasi-projection: project using relatively large penalty tolerance, i.e. to the neighborhood of F

Example of the REPA Results, Large F



Example of the REPA Results, Small F



Constrained Optimization with Evolutionary Algorithms Test Problems

synthetic problems **g01–g13**, real-life design problems **tens, vess**

Problem	Difficulty	n	Obj. function	ρ	LI	NI	LE	NE
g01	D	13	quadratic	0.0003	9	0	0	0
g02	D	20	nonlinear	99.9973	2	0	0	0
g03	D	10	nonlinear	0.0026	0	0	0	1
g04	A	5	quadratic	27.0079	4	2	0	0
g05	VD	4	nonlinear	0.0000	2	0	0	3
g06	A	2	nonlinear	0.0057	0	2	0	0
g07	A	10	quadratic	0.0000	3	5	0	0
g08	E	2	nonlinear	0.8581	0	2	0	0
g09	A	7	nonlinear	0.5199	0	4	0	0
g10	D	8	linear	0.0020	6	0	0	0
g11	E	2	quadratic	0.0973	0	0	0	1
g12	E	3	quadratic	4.7697	0	9^3	0	0
g13	VD	5	nonlinear	0.0000	0	0	1	2
vess	A	4	quadratic	39.6762	3	1	0	0
tens	E	3	quadratic	0.7537	1	3	0	0

Studies on Constraints Projection: Methodology

- Constraints from the test problems set, plus one

$$g_1(\mathbf{x}) = x_1^2 + x_2^2 - 1.1^2 = 0$$
$$h_1(\mathbf{x}) = x_1 - 1 \leq 0$$
$$h_2(\mathbf{x}) = -x_1 - 1 \leq 0$$
$$h_3(\mathbf{x}) = x_2 - 1 \leq 0$$
$$h_4(\mathbf{x}) = -x_2 - 1 \leq 0$$

- Projection methods from COSY Infinity (SIMPLEX, LMDIF, ANNEALING), combined methods are combinations of standard methods
- Combinations of power penalty functions $a = 0, 1, 2$
- Initial points generated randomly, uniformly distributed over $S = [-100, 100]^v$ and $S = [-1000, 1000]^v$
- For all methods the maximum number of steps is 1000, precision is 10^{-5}
- Ranking by success rate with some emphasis put on the average number of steps

Studies on Constraints Projection: Results, $S = [-100, 100]^v$

#	I			II			III		
	method	% succ	steps	method	% succ	steps	method	% succ	steps
0	L+A(z, z)	100.0	82	L(z, z)	98.1	50	S+L(z, z)	100.0	70
1	L(z)	100.0	45	L:c(z)	98.67	94	L(z^2)	100.0	234
2	L(z)	97.8	65	S+A:c(z^2)	97.6	93	S+A:c(z)	97.0	94
3	S(z)	100.0	258	L+A(z)	100.0	270	S+L(z)	100.0	333
4	L(z)	100.0	19	L+A(z)	100.0	53	L(z^2)	100.0	80
5	L($z^2 + z$)	10.1	938	-	-	-	-	-	-
6	L(z)	99.9	83	L(z^2)	99.6	121	S+L(z)	100.0	191
7	L(z)	100.0	122	L(z^2)	99.3	342	-	-	-
8	L+A(z)	100.0	66	S+L(z)	100.0	67	L(z)	99.5	56
9	S:c(z^2)	96.1	327	L+A(z^2)	97.5	513	L+A(z)	89.6	373
10	L(z^2)	81.9	386	S+L(z^2)	76.0	501	L+A(z)	74.1	379
11	L(z)	100.0	20	S+L(z)	100.0	50	L+A(z)	100.0	56
12	S+L(z)	100.0	125	L+A(z)	100.0	132	S+L(z^2)	100.0	210
13	L+A(z)	99.9	361	S+L(z)	98.3	327	L(z)	75.6	342
pres	S+L:c(z^2)	98.3	242	L+A(z)	91.6	141	L(z)	89.4	90
tens	L(z^2)	22.8	202	L(z)	20.7	329	S+A:c(z^2)	25.1	902

Studies on Constraints Projection: Results, $S = [-100, 100]^v$

#	I			II			III		
	method	% succ	steps	method	% succ	steps	method	% succ	steps
0	L+A(z, z)	100.0	82	L(z, z)	98.1	50	S+L(z, z)	100.0	70
1	L(z)	100.0	45	L:c(z)	98.67	94	L(z^2)	100.0	234
2	L(z)	97.8	65	S+A:c(z^2)	97.6	93	S+A:c(z)	97.0	94
3	S(z)	100.0	258	L+A(z)	100.0	270	S+L(z)	100.0	333
4	L(z)	100.0	19	L+A(z)	100.0	53	L(z^2)	100.0	80
5	L($z^2 + z$)	10.1	938	-	-	-	-	-	-
6	L(z)	99.9	83	L(z^2)	99.6	121	S+L(z)	100.0	191
7	L(z)	100.0	122	L(z^2)	99.3	342	-	-	-
8	L+A(z)	100.0	66	S+L(z)	100.0	67	L(z)	99.5	56
9	S:c(z^2)	96.1	327	L+A(z^2)	97.5	513	L+A(z)	89.6	373
10	L(z^2)	81.9	386	S+L(z^2)	76.0	501	L+A(z)	74.1	379
11	L(z)	100.0	20	S+L(z)	100.0	50	L+A(z)	100.0	56
12	S+L(z)	100.0	125	L+A(z)	100.0	132	S+L(z^2)	100.0	210
13	L+A(z)	99.9	361	S+L(z)	98.3	327	L(z)	75.6	342
pres	S+L:c(z^2)	98.3	242	L+A(z)	91.6	141	L(z)	89.4	90
tens	L(z^2)	22.8	202	L(z)	20.7	329	S+A:c(z^2)	25.1	902

Studies on Constraints Projection: Results, $S = [-100, 100]^v$

#	I			II			III		
	method	% succ	steps	method	% succ	steps	method	% succ	steps
0	L+A(z, z)	100.0	82	L(z, z)	98.1	50	S+L(z, z)	100.0	70
1	L(z)	100.0	45	L:c(z)	98.67	94	L(z^2)	100.0	234
2	L(z)	97.8	65	S+A:c(z^2)	97.6	93	S+A:c(z)	97.0	94
3	S(z)	100.0	258	L+A(z)	100.0	270	S+L(z)	100.0	333
4	L(z)	100.0	19	L+A(z)	100.0	53	L(z^2)	100.0	80
5	L($z^2 + z$)	10.1	938	-	-	-	-	-	-
6	L(z)	99.9	83	L(z^2)	99.6	121	S+L(z)	100.0	191
7	L(z)	100.0	122	L(z^2)	99.3	342	-	-	-
8	L+A(z)	100.0	66	S+L(z)	100.0	67	L(z)	99.5	56
9	S:c(z^2)	96.1	327	L+A(z^2)	97.5	513	L+A(z)	89.6	373
10	L(z^2)	81.9	386	S+L(z^2)	76.0	501	L+A(z)	74.1	379
11	L(z)	100.0	20	S+L(z)	100.0	50	L+A(z)	100.0	56
12	S+L(z)	100.0	125	L+A(z)	100.0	132	S+L(z^2)	100.0	210
13	L+A(z)	99.9	361	S+L(z)	98.3	327	L(z)	75.6	342
pres	S+L:c(z^2)	98.3	242	L+A(z)	91.6	141	L(z)	89.4	90
tens	L(z^2)	22.8	202	L(z)	20.7	329	S+A:c(z^2)	25.1	902

Studies on Constraints Projection: Results, $S = [-1000, 1000]^v$

#	I			II			III		
	method	% succ	steps	method	% succ	steps	method	% succ	steps
0	L+A(z, z)	100.0	106	L(z, z)	99.6	45	S+L(z, z)	100.0	90
1	L(z)	100.0	45	L:c(z)	98.5	97	L(z^2)	100.0	278
2	L(z)	94.0	103	S+A:c(z)	78.8	302	S+A:c(z^2)	78.6	301
3	S(z)	99.5	343	S+L(z)	99.9	466	L+A(z)	100.0	419
4	L(z)	99.9	41	L+A(z)	100.0	130	L(z^2)	100.0	125
5	-	-	-	-	-	-	-	-	-
6	L(z)	99.5	116	L(z^2)	98.4	183	S+L(z)	100.0	209
7	L(z)	100.0	129	L(z^2)	97.2	514	-	-	-
8	L+A(z)	100.0	92	S+L(z)	100.0	91	L(z)	98.1	80
9	S:c(z^2)	59.3	715	L+A(z^2)	47.4	572	L+A(z)	25.4	913
10	L(z^2)	77.8	445	S+L(z^2)	74.3	540	L+A(z)	66.6	453
11	L(z)	99.9	25	S+L(z)	99.9	69	L+A(z)	100.0	79
12	S+L(z)	100.0	171	L+A(z)	100.0	187	S+L(z^2)	100.0	295
13	L+A(z)	98.3	472	S+L(z)	98.1	502	L(z)	66.6	542
pres	S+L:c(z^2)	93.5	268	S:c(z^2)	93.3	123	S:c(z)	92.1	121
tens	L(z^2)	4.6	196	L(z)	2.5	168	S+A:c(z^2)	15.3	984

Studies on Constraints Projection: Results, $S = [-1000, 1000]^v$

#	I			II			III		
	method	% succ	steps	method	% succ	steps	method	% succ	steps
0	L+A(z, z)	100.0	106	L(z, z)	99.6	45	S+L(z, z)	100.0	90
1	L(z)	100.0	45	L:c(z)	98.5	97	L(z^2)	100.0	278
2	L(z)	94.0	103	S+A:c(z)	78.8	302	S+A:c(z^2)	78.6	301
3	S(z)	99.5	343	S+L(z)	99.9	466	L+A(z)	100.0	419
4	L(z)	99.9	41	L+A(z)	100.0	130	L(z^2)	100.0	125
5	-	-	-	-	-	-	-	-	-
6	L(z)	99.5	116	L(z^2)	98.4	183	S+L(z)	100.0	209
7	L(z)	100.0	129	L(z^2)	97.2	514	-	-	-
8	L+A(z)	100.0	92	S+L(z)	100.0	91	L(z)	98.1	80
9	S:c(z^2)	59.3	715	L+A(z^2)	47.4	572	L+A(z)	25.4	913
10	L(z^2)	77.8	445	S+L(z^2)	74.3	540	L+A(z)	66.6	453
11	L(z)	99.9	25	S+L(z)	99.9	69	L+A(z)	100.0	79
12	S+L(z)	100.0	171	L+A(z)	100.0	187	S+L(z^2)	100.0	295
13	L+A(z)	98.3	472	S+L(z)	98.1	502	L(z)	66.6	542
pres	S+L:c(z^2)	93.5	268	S:c(z^2)	93.3	123	S:c(z)	92.1	121
tens	L(z^2)	4.6	196	L(z)	2.5	168	S+A:c(z^2)	15.3	984

Studies on Constraints Projection: Results, $S = [-1000, 1000]^v$

#	I			II			III		
	method	% succ	steps	method	% succ	steps	method	% succ	steps
0	L+A(z, z)	100.0	106	L(z, z)	99.6	45	S+L(z, z)	100.0	90
1	L(z)	100.0	45	L:c(z)	98.5	97	L(z^2)	100.0	278
2	L(z)	94.0	103	S+A:c(z)	78.8	302	S+A:c(z^2)	78.6	301
3	S(z)	99.5	343	S+L(z)	99.9	466	L+A(z)	100.0	419
4	L(z)	99.9	41	L+A(z)	100.0	130	L(z^2)	100.0	125
5	-	-	-	-	-	-	-	-	-
6	L(z)	99.5	116	L(z^2)	98.4	183	S+L(z)	100.0	209
7	L(z)	100.0	129	L(z^2)	97.2	514	-	-	-
8	L+A(z)	100.0	92	S+L(z)	100.0	91	L(z)	98.1	80
9	S:c(z^2)	59.3	715	L+A(z^2)	47.4	572	L+A(z)	25.4	913
10	L(z^2)	77.8	445	S+L(z^2)	74.3	540	L+A(z)	66.6	453
11	L(z)	99.9	25	S+L(z)	99.9	69	L+A(z)	100.0	79
12	S+L(z)	100.0	171	L+A(z)	100.0	187	S+L(z^2)	100.0	295
13	L+A(z)	98.3	472	S+L(z)	98.1	502	L(z)	66.6	542
pres	S+L:c(z^2)	93.5	268	S:c(z^2)	93.3	123	S:c(z)	92.1	121
tens	L(z^2)	4.6	196	L(z)	2.5	168	S+A:c(z^2)	15.3	984

Percent Successful Runs, Different Methods

Problem	Diff.	v	n	Success Rate (%)			
				Killing	Killing+Penalty	Anneal. Penalty	REPA
G01	D	13	9	2	3	100	9
G02	D	20	2	100	100	100	100
G03	D	10	1	3	2	100	100
G04	A	5	6	100	100	99	100
G05	VD	4	5	0	0	0	100
G06	A	2	2	23	2	54	99
G07	A	10	8	0	0	100	100
G08	E	2	2	100	100	100	100
G09	A	7	4	100	100	100	100
G10	D	8	6	10	0	0	11
G11	E	2	1	12	1	100	99
G12	E	3	1	100	95	100	100
G13	VD	5	3	0	0	76	100
tens	E	3	4	96	44	89	100
vess	A	4	4	100	100	100	100

Summary of the performance, REPA method, after 150 generations

Prob.	Optimum	Best	Median	Mean	Worst
G01	-15	-14.407890	-14.120216	-13.277590	-6.673952
G02	-0.803619	-0.780622	-0.698852	-0.694653	-0.583719
G03	-1	-0.987591	-0.9559559	-0.9661996	-0.378451
G04	-30665.539	-30663.677834	-30625.175701	-30619.883212	-30511.318
G05	5126.4981	**5126.498109**	5126.517730	5126.67221	5130.978
G06	-6961.81388	-6961.830259	-6601.428949	-6111.785535	-3531.262
G07	24.3062091	25.664348	28.512014	28.804470	35.3144229
G08	-0.095825	**-0.0958250**	-0.09582496	-0.09311891	-0.0291434
G09	680.6300573	680.8126323	681.5472870	681.768380	685.1725065
G10	7049.3307	7097.356559	8713.695245	9080.98370	11245.061
G11	0.75	**0.7500003**	0.750788	0.7551577	0.8292849
G12	-1	**-0.999999**	-0.999999	-0.9999998	-0.999996
G13	0.0539498	**0.05395041**	0.05398875	0.05409692	0.05900387
tens	0.012681	**0.01268532**	0.013211	0.01546248	0.1070929
vess	6059.946341	8825.1065735	10004.415854	11346.495914	40395.1935

- G01: high-dimensional (13) and has the largest number of constraint functions (9)
- Decreasing the projection penalty tolerance to 1 (from default 10^{-5}) and increasing the maximum allowed number of steps for projection to 70 (from default 50) we can restore the success rate up to 100% and increase the quality of results to

```
-14.957892
-14.371071
-14.327610
-13.125392
```

Conclusions

- REPA method has performance that is comparable to the one of existing methods
- On test problems G05, G13 considered VERY DIFFICULT it shows superior performance
- Method is not tied to a particular flavour of EA, can be easily extended and modified for the problem
- However... large number of parameters $\Rightarrow$ flexibility for the price of possibly expensive fine-tuning
- For the standard test problems set performance of REPROPT is assessed, default parameters selected
- Future directions:
  - More tests
  - Integration with COSY-GO
  - Extensions: other optimizers for projectoin, feasible elitism (REFIND much less expensive!)